

PN - JP2001099925 A20010413
 PD - 2001-04-13
 PR - JP19990280187 19990930
 OPD - 1999-09-30
 TI - PARALLEL MULTI-TARGET TRACKING DEVICE
 IN - TAKAHASHI KATSUMI SATO HIROYUKI
 PA - MITSUBISHI ELECTRIC CORP
 IC - G01S13/66 ; G06F9/46 ; G06F15/177

© WPI / DERWENT

TI - Multiple goal tracking apparatus used in radar, updates object information based on process result of search tree node and inputs to estimate area generator

PR - JP19990280187 19990930

PN - JP2001099925 A20010413 DW200138 G01S13/66 011pp

PA - (MITQ) MITSUBISHI ELECTRIC CORP

IC - G01S13/66 ; G06F9/46 ; G06F15/177

AB - JP2001099925 NOVELTY - A generator (3) generates cluster from estimate area generated by generator (2), and obtains search tree node about cluster. Task processors (5a-5d) process each node divided by divider (4), using processes that are assigned dynamically with respect to each node, based on load situation of each processor. Updating unit (6) updates object information based on the process result, and outputs to generator (2).

- USE - Joint probabilistic data association (JPDA) based multiple goal tracking apparatus used in radar.
- ADVANTAGE - Performs optimum load distribution to each task processors.
- DESCRIPTION OF DRAWING(S) - The figure shows the block diagram of the components of tracking apparatus. (Drawing includes non-English text).
- Estimate area generator 2
- Cluster area generator 3
- Divider 4
- Task processors 5a-5d
- Updating unit 6
- (Dwg.1/6)

OPD - 1999-09-30
 AN - 2001-360819 [38]

© PAJ / JPO

PN - JP2001099925 A20010413
 PD - 2001-04-13
 AP - JP19990280187 19990930

- IN - SATO HIROYUKI;TAKAHASHI KATSUMI
- PA - MITSUBISHI ELECTRIC CORP
- TI - PARALLEL MULTI-TARGET TRACKING DEVICE
- AB - PROBLEM TO BE SOLVED: To obtain a parallel multi-target tracking device capable of conducting optimum load dispersion in parallel multi-target tracking processing even in a condition where another processing is executed on a processor for executing the processing or where another communication is carried out on a network.
- SOLUTION: As to a cluster generated from observation point information and predicted areas by a cluster generating part 3, when the nodes of a search tree are generated by a node dividing/integrating part 4 to be distributed plural task processing parts 5a-5d to be processed, the distribution of the processing to each of the respective task processing parts 5a-5d is assigned dynamically in response to a load condition in each of the task processing parts 5a-5d by a load dispersing part 9.
- G01S13/66 ;G06F9/46 ;G06F15/177

PATENT ABSTRACTS OF JAPAN

(21) Application number: 11280187

(51) Intl. Cl.: G01S 13/66 G06F 9/46 G06F 15/177

(22) Application date: 30.09.99

(30) Priority:

(43) Date of application
publication: 13.04.01(84) Designated contracting
states:

(71) Applicant: MITSUBISHI ELECTRIC CORP

(72) Inventor: SATO HIROYUKI
TAKAHASHI KATSUMI

(74) Representative:

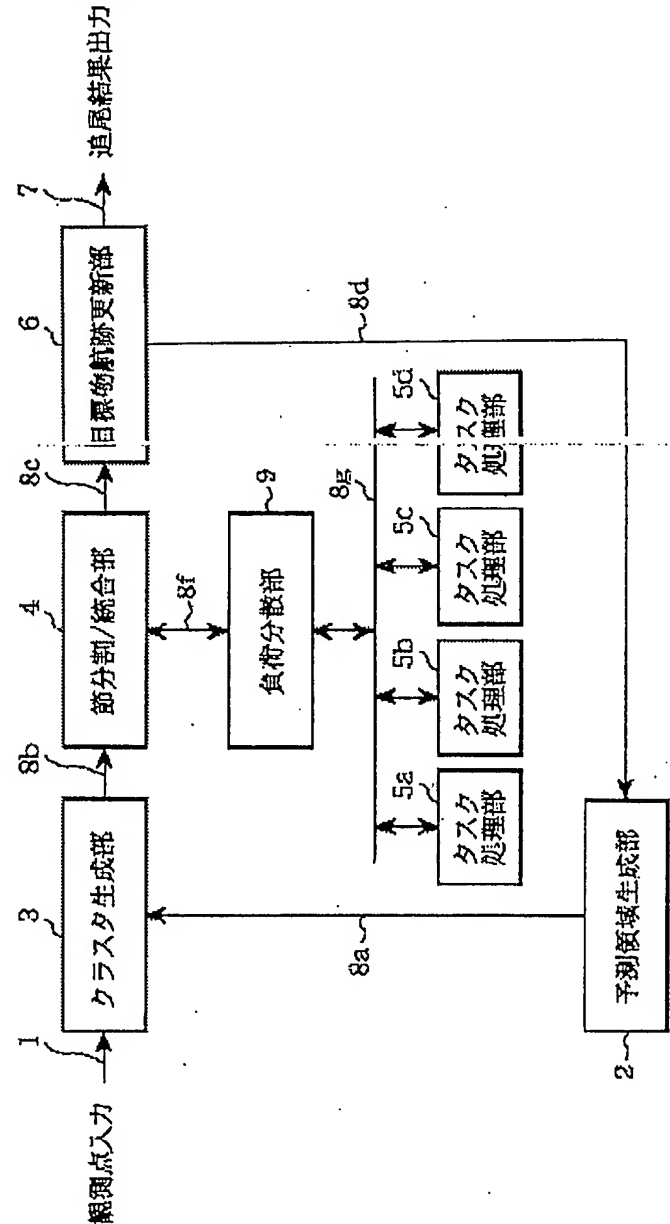
(54) PARALLEL MULTI-TARGET TRACKING DEVICE

(57) Abstract:

PROBLEM TO BE SOLVED: To obtain a parallel multi-target tracking device capable of conducting optimum load dispersion in parallel multi-target tracking processing even in a condition where another processing is executed on a processor for executing the processing or where another communication is carried out on a network.

SOLUTION: As to a cluster generated from observation point information and predicted areas by a cluster generating part 3, when the nodes of a search tree are generated by a node dividing/integrating part 4 to be distributed plural task processing parts 5a-5d to be processed, the distribution of the processing to each of the respective task processing parts 5a-5d is assigned dynamically in response to a load condition in each of the task processing parts 5a-5d by a load dispersing part 9.

COPYRIGHT: (C)2001,JPO



(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2001-99925

(P2001-99925A)

(43) 公開日 平成13年4月13日 (2001.4.13)

(51) Int.Cl. ⁷	識別記号	F I	テ-マ-ト (参考)
G 0 1 S 13/66		G 0 1 S 13/66	5 B 0 4 5
G 0 6 F 9/46	3 6 0	G 0 6 F 9/46	3 6 0 C 5 B 0 9 8
15/177	6 7 4	15/177	6 7 4 A 5 J 0 7 0

審査請求 未請求 請求項の数 8 O L (全 11 頁)

(21) 出願番号 特願平11-280187

(22) 出願日 平成11年9月30日 (1999.9.30)

(71) 出願人 000006013

三菱電機株式会社

東京都千代田区丸の内二丁目2番3号

(72) 発明者 佐藤 裕幸

東京都千代田区丸の内二丁目2番3号 三
菱電機株式会社内

(72) 発明者 高橋 勝己

東京都千代田区丸の内二丁目2番3号 三
菱電機株式会社内

(74) 代理人 100066474

弁理士 田澤 博昭 (外1名)

Fターム(参考) 5B045 AA00 GC02 GG04

5B098 AA10 GC01 GC08 GD02 GD14

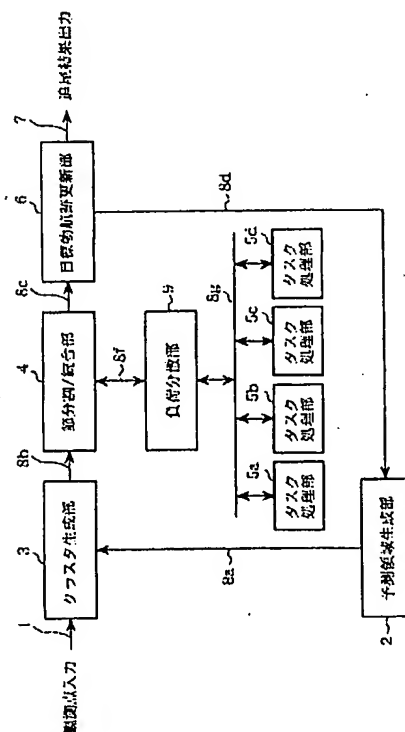
5J070 AK40 AL01 BB04 BB06

(54) 【発明の名称】 並列多目標追尾装置

(57) 【要約】

【課題】 処理を実行するプロセッサ上で他の処理が実行されていたり、ネットワーク上で他の通信がなされている状態であっても、並列多目標追尾処理において、最適な負荷分散を行うことが可能な並列多目標追尾装置を得る。

【解決手段】 クラスタ生成部3が観測点情報と予測領域より生成したクラスタについて、節分割/統合部4にて探索木の節を生成し、それをタスクとして複数のタスク処理部5a～5dに分配して処理する際に、各タスク処理部への節の処理の分配を負荷分散部9によって、各タスク処理部の負荷状況に応じて動的に割りあたるようにした。



【特許請求の範囲】

【請求項1】 複数のタスク処理部に対する負荷を分散させて、複数の目標物の追尾を行う並列多目標追尾装置において、

前記目標物の移動を予測し、その予測結果に応じて予測領域を生成する予測領域生成部と、

入力された観測点情報と、前記予測領域生成部にて生成された予測領域からクラスタを生成するクラスタ生成部と、

前記クラスタ生成部にて生成されたクラスタについて探索木の節を生成し、それをタスクとして分割・統合する節分割・統合部と、

前記節分割・統合部によって分割された個々の節について、その処理を行う複数のタスク処理部と、

前記節分割・統合部によって分割された節に対する処理を、前記各タスク処理部の負荷状況に基づいて動的に割り当てる負荷分散部と、

前記節分割・統合部の処理結果に基づいて前記目標物の情報を更新し、それを追尾結果として出力するとともに、前記目標物の移動予測のために前記予測領域生成部に入力する目標物航跡更新部とを備えたことを特徴とする並列多目標追尾装置。

【請求項2】 負荷分散部が、最初は複数のタスク処理部の負荷が均等になるように、各タスク処理部に節の処理を分配し、

その後、定期的に前記各タスク処理部の負荷状況を監視して、負荷の多いタスク処理部から負荷の少なくなったタスク処理部に節の処理を移動させることにより、各タスク処理部の負荷を均等化しながら負荷分散を行うことを特徴とする請求項1記載の並列多目標追尾装置。

【請求項3】 負荷分散部が、節を処理する各タスク処理部に対して1単位ずつ節の処理を分配し、

前記各タスク処理部は、分配された節の処理が終了すると、次の節の分配を負荷分散部へ要求し、

前記負荷分散部は次の節の分配要求を受けると、当該節の分配を要求したタスク処理部に対して、1単位の節を分配することを特徴とする請求項1記載の並列多目標追尾装置。

【請求項4】 負荷分散部が、節を処理する各タスク処理部に対して、最初に複数単位ずつの節の処理を分配し、

前記各タスク処理部は、前記負荷分散部にて分配された複数単位の節を1単位を除いて保持して、残った1単位の節の処理を行い、当該節の処理が終了すると、負荷分散部に対して次の節の分配を要求するとともに、その要求に対する応答が返される前に保持しておいた次の1単位の節の処理を開始し、

前記負荷分散部は次の節の分配要求を受けると、当該節の分配を要求したタスク処理部に対して次の節の分配を行うことを特徴とする請求項1記載の並列多目標追尾装

置。

【請求項5】 負荷分散の単位を、時間とともに小さくしていくことを特徴とする請求項1記載の並列多目標追尾装置。

【請求項6】 タスク処理部において1つの負荷分散単位の処理時間が一定時間以上長くなった場合に、当該タスク処理部が処理している節の下位の一部の節群の処理を、負荷分散部へ返すことを特徴とする請求項1記載の並列多目標追尾装置。

【請求項7】 タスク処理部が負荷分散部へ節群の処理を返すと判断する1つの負荷分散単位の処理時間の上限値を、時間とともに小さくしていくことを特徴とする請求項6記載の並列多目標追尾装置。

【請求項8】 複数のタスク処理部の内の1つを、負荷分散部と同一のプロセッサ上で動作させることを特徴とする請求項1記載の並列多目標追尾装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】この発明は、レーダにおける複数目標物の追尾手法として、JPDA (Joint Probabilistic Data Association) のような複数の目標物の組み合わせを想定したものを採用し、処理を複数のタスク処理部に動的に割り当てて負荷分散をはかった並列多目標追尾装置に関するものである。

【0002】

【従来の技術】多目標追尾を並列処理する従来の追尾方法としては、例えば、処理を機能分割によって負荷分散する "On Mapping a Tracking Algorithm Onto Parallel Processors" (IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS VOL. 26, NO. 5, 1990) に示されたもの等がある。図5および図6は従来の並列多目標追尾装置による追尾方法を示す説明図であり、追尾処理とプロセッサをグラフとして示したものである。この追尾方法では、図5のように処理をタスクグラフとして表現し、各タスクの負荷を情報の流れに基づいて、プロセッサへの分割を行なう。

【0003】図5は処理の関係を示すタスクグラフである。タスクグラフとは、処理の相関関係を示すグラフであり、丸印によって示される(1)～(8)の処理と、それら各処理(1)～(8)の繋がりを示す矢印110～119とによって表現されている。なお、処理(1)～(8)は計算機などで行なわれる処理自体を示しており、ここでは、タスク(1)101～タスク(8)108としている。また、矢印110～119はこのタスク(1)101～タスク(8)108における計算結果が利用される先を示している。

【0004】すなわち、例えば、

$C = A + B$ …… (あ)

$E = C + D$ …… (い)

という2つの処理(あ)および(い)があった場合、

(あ)→(い)という矢印で表す。なお、ここで、

“(あ)”および“(い)”は、“あ”および“い”を丸で囲んだ状態を示すものとする。

【0005】タスクグラフで表した処理は、全て、バラバラに複数のプロセッサに分割して処理させることができる。先の例でいえば、処理(あ)の計算と処理(い)の計算とをそれぞれ別のプロセッサで行なうことができる。この場合、処理(い)の計算を行うプロセッサは、処理(あ)の計算結果を処理(あ)の計算を行うプロセッサから受け取ることになる。このように、図5に示したタスクグラフにおいて、矢印110～119で繋がっているタスク(1)101～タスク(8)108は計算結果のやりとりが必要となる。

【0006】タスクグラフには当然、矢印110～119で繋がったものと繋がっていないものとが存在するが、矢印110～119で繋がっていないもの同士はお互いにデータをやりとりする必要はない。図5に示す例でいえば、タスク(1)101とタスク(2)102～タスク(5)105は矢印110、113、115、117を介して直接繋がっており、タスク(2)102には矢印111を介してタスク(6)106が、タスク(5)105には矢印118を介してタスク(7)107がそれぞれ繋がっている。また、タスク(6)106、タスク(3)103、タスク(4)104、タスク(7)107とタスク(8)108は、それぞれ矢印112、114、116、119を介して直接繋がっている。従って、矢印で繋がっていないペアは、タスク(2)102とタスク(5)105のペア、およびタスク(6)106とタスク(7)107のペアである。

【0007】一方、タスク(1)101とタスク(2)102のペアや、タスク(2)102とタスク(6)106のペア、およびタスク(5)105とタスク(7)107のペア等は矢印110、111あるいは118で繋がっており、例えば、タスク(1)101とタスク(2)102なら、タスク(1)101の計算が終わらない限りタスク(2)102の計算も終わらないという関係になっている。

【0008】そのとき、以下に示した2通りの分け方があった場合、後者の分け方を選択するようにする。それは、前者の分け方ではプロセッサ間で計算結果を渡す必要が生じるが、後者ではその必要が生じないからである。

【0009】タスク(2)102とタスク(5)105を1つのプロセッサに割り当て、タスク(6)106とタスク(7)107を他の1つのプロセッサに割り当てる。

タスク(2)102とタスク(6)106を1つの

プロセッサに割り当て、タスク(5)105とタスク(7)107を他の1つのプロセッサに割り当てる。

【0010】さらに、各タスク101～108には通常、必要な計算量が示されている(図5では省略)。従って、タスクグラフに与えられている処理を複数のプロセッサに分ける場合、以下の2つの方針に基づいて行うことになる。

【0011】各プロセッサに分けたタスクの必要計算量の合計がなるべく同じになるように割り当てる。

【0012】矢印110～119で繋がっているタスク101～108同士は、同じプロセッサに割り当てる。

【0013】図6はプロセッサグラフであり、4つのプロセッサ(P1)121～(P2)124が環状に繋がっている例を示している。すなわち、プロセッサ(P1)121にプロセッサ(P2)122が繋がりが、プロセッサ(P2)122にプロセッサ(P3)123が繋がりが、プロセッサ(P3)123にプロセッサ(P4)124が繋がりが、プロセッサ(P4)124にプロセッサ(P1)121が繋がっている。

【0014】このように矢印110～119で繋がっているタスク101～108同士を同じプロセッサ121～124に割り当てるとというのが、従来の並列多目標追尾装置で用いられている追尾方法である。この図5に示されているタスクグラフで、丸で表現されているタスク101～108は機能的に一塊となる処理である。JPDA等のように、目標物と観測点の組合せを考慮して追尾処理を行う場合、対象とする問題が大きくなると、目標物と観測点との組合せを求めるタスクは、他のタスクに比べて、処理の負荷が非常に大きくなる。

【0015】

【発明が解決しようとする課題】従来の並列多目標追尾装置は以上のように構成されているので、JPDAのように複数の目標物と観測点の組合せを考慮して追尾処理を行なう方式では、目標物と観測点の組合せを求めるための特定の1つのタスク101～108の処理の負荷が特別に非常に大きくなってしまふことがあり、その場合には、上述したように複数のプロセッサ121～124を用いて各タスク101～108の並列処理を行っても、負荷が特別大きくなってしまふタスク101～108の処理を行ったプロセッサ121～124だけは処理が終わらず、結果として、全体の処理が完全に終わるまでに長時間を要し、複数のプロセッサ121～124を用いた負荷分散効果が得られず、並列処理を行っている意味がなくなってしまうという課題があった。

【0016】目標物と観測点の組合せを求めるためのタスク101～108を処理負荷ができるだけ均等になるように分割し、それぞれを別プロセッサ121～124で処理することは可能である。しかしながら、各プロセッサ121～124が汎用のワークステーションで動作する場合、目標物と観測点の組合せを求めるタスク10

1~108以外の他の処理や他のユーザの処理も、同一のプロセッサ121~124上で実行される可能性があり、それら他の処理が各プロセッサ121~124で一定になるとは限らず、各プロセッサ121~124で処理されるタスク101~108の実行時間も均等にはならず、最適な負荷分散とはならなくなってしまうという課題があった。

【0017】また、各プロセッサ（ワークステーション）121~124では、目標物と観測点の組合せを求めるためのタスク101~108以外の処理は行わないことにしても、各プロセッサ121~124が汎用のネットワークで接続されている場合は、タスク101~108の処理を行うプロセッサ121~124以外のプロセッサも、同じネットワークを共有して使用していることが考えられ、そのような場合には、プロセッサ121~124間の通信時間が一定にはならないので、各プロセッサ121~124でのタスク101~108の処理の終了が一定とはならず、最適な負荷分散とはならなくなってしまうという課題があった。

【0018】この発明は上記のような課題を解決するためになされたもので、目標物と観測点の組合せを求めるタスク等の特別高負荷となってしまうタスクに対して負荷分散を行い、タスクを実行するプロセッサ上で他の処理が動作していたり、ネットワーク上で他の通信がなされている状態であっても、並列多目標追尾処理において、最適な負荷分散を行なうことが可能な並列多目標追尾装置を得ることを目的とする。

【0019】

【課題を解決するための手段】この発明に係る並列多目標追尾装置は、クラスタ生成部が観測点情報と予測領域より生成したクラスタについて、節分割・統合部にて探索木の節を生成し、それをタスクとして複数のタスク処理部に分配して処理する際に、負荷分散部によって、各タスク処理部への節の処理の分配を、各タスク処理部の負荷状況に応じて行うようにしたものである。

【0020】この発明に係る並列多目標追尾装置は、各タスク処理部の負荷状況を定期的に監視して、タスク処理部の負荷が少なくなると、負荷の多いタスク処理部から負荷を移動させることにより、各タスク処理部における負荷を均等化するようにしたものである。

【0021】この発明に係る並列多目標追尾装置は、各タスク処理部に対して1単位ずつ節の処理を分配し、分配された節の処理が終了したタスク処理部は、次の節の分配を要求して新たな1単位の節の処理の分配を受けることで負荷分散を行うようにしたものである。

【0022】この発明に係る並列多目標追尾装置は、負荷分散部より各タスク処理部に対して、最初に複数単位ずつの節の処理を分配し、各タスク処理部では受け取った節の内の1単位の節の処理を開始して、残りを保持しておき、当該節の処理が終了すると、次の節の分配を負

荷分散部に要求するとともに、その要求に対する応答が返される前に保持しておいた次の1単位の節の処理を開始し、分配要求を受けた負荷分散部は、節の分配を要求したタスク処理部に対して次の節の分配を行うようにしたものである。

【0023】この発明に係る並列多目標追尾装置は、時間とともに負荷分散の単位を小さくしていくようにしたものである。

【0024】この発明に係る並列多目標追尾装置は、1つの負荷分散単位の処理時間が一定時間以上になると、処理している節の下の一部の節群の処理を負荷分散部へ返すようにしたものである。

【0025】この発明に係る並列多目標追尾装置は、一部の節群の処理を負荷分散部へ返すと判断する際の処理時間の上限値を、時間とともに小さくしていくようにしたものである。

【0026】この発明に係る並列多目標追尾装置は、負荷分散部と同一のプロセッサ上でタスク処理部の1つを動作させるようにしたものである。

【0027】

【発明の実施の形態】以下、この発明の実施の一形態を説明する。

実施の形態1. 図1はこの発明の実施の形態1による並列多目標追尾装置の構成を示すブロック図である。図において、1は外部からの観測点情報の入力が行われる入力線である。2は目標物の移動を予測し、その予測結果に応じて予測領域を生成する予測領域生成部である。3は入力線1から入力された観測点情報と、予測領域生成部2の生成した予測領域からクラスタを生成するクラスタ生成部である。4はこのクラスタ生成部3で生成されたクラスタについて探索木の節を生成し、これをタスクとして分割・統合を行う節分割・統合部である。5a~5dは節分割・統合部4で分割された個々のタスク

（節）についての処理を行うタスク処理部である。6は節分割・統合部4で統合されたタスク処理部5a~5dの処理結果に基づいて、目標物の持つ情報（航跡）の更新を行う目標物航跡更新部である。7は目標物航跡更新部6で更新された情報を追尾結果として外部に出力する出力線である。8a~8dはこれら各部の処理結果の転送に用いる転送線である。

【0028】また、9は節分割・統合部4によって分割された節に対する処理を、各タスク処理部5a~5dに対して、それぞれの負荷状況に基づいて分散して割り当てる負荷分散部である。8fはこの負荷分散部9と節分割・統合部4との間を結ぶ転送線であり、8gは負荷分散部9と各タスク処理部5a~5dとの間を結ぶ転送線である。

【0029】次に動作について説明する。クラスタ生成部3には、外部からの観測点情報が入力線1を介して、また予測領域生成部2の作成した予測領域が転送路8a

を介してそれぞれ入力される。これら観測点情報および予測領域を受けたクラスタ生成部3は、複数の目標物の予測領域が重なり、かつその重なった領域に観測点が存在する時には、その予測領域を結合して1つのクラスタを生成する。このクラスタ生成部3で生成された1つのクラスタの情報(観測点と目標物の予測領域)は、転送路8bを介して節分割/統合部4に送られ、節分割/統合部4は受け取ったクラスタの情報から探索木の節を生成する。

【0030】負荷分散部9はこの節分割/統合部4によって生成された節を転送路8fを介して取り込み、各タスク処理部5a~5dの負荷状況に応じて、その節の処理を転送路8gを介して各タスク処理部5a~5dに分配する。各タスク処理部5a~5dは受け取った節から、探索木の葉(追尾の組合せの1つに相当する)を求め、その処理結果を転送路8gおよび8fを介して節分割/統合部4へ送る。節分割/統合部4は送られてきた各タスク処理部5a~5dの処理結果を統合し、それを転送路8cを介して目標物航跡更新部6に送る。目標物航跡更新部6は目標物の情報を持ち、送られてきた個々の組合せの可能性をもとに、目標物の情報を更新する。この目標物航跡更新部6にて更新された目標物の情報は、出力線7を介して外部に出力されるとともに、転送線8dを介して予測領域生成部2へも送られる。予測領域生成部2はこの目標物航跡更新部6で更新された目標物の情報に基づいて各目標物について移動予測を行い、その予測領域を生成して転送線8aよりクラスタ生成部3に送る。この装置は、以上の動作を繰り返すことによって、動的負荷分散を用いた並列多目標追尾装置として動作する。

【0031】以下、JPDAを用いる場合を例に、各部の動作を説明する。ここで、図2はJPDAにおける目標物と観測点の関係を示す説明図、図3はこの目標物と観測点の関係を表した行列Ωを示す説明図、図4は目標物と観測点の組合せを表した探索木を示す説明図である。図2において、 $t(1)$ および $t(2)$ は目標物であり、黒丸で示したこの目標物 $t(1)$ 、 $t(2)$ の予測座標を中心とする楕円は予測範囲である。また、 $y(1) \sim y(4)$ はそれぞれ観測点である。

【0032】なお、図3に示す行列Ωはそれぞれ、縦が目標物 $t(0) \sim t(2)$ 、横が観測点 $y(1) \sim y(4)$ で分かれており、目標物 $t(0)$ は対応する目標なしを示している。また、行列Ω内の“1”は可能性ありを、“0”は可能性なしをそれぞれ示しており、この行列Ωは次の内容を表したものである。

- ・ 観測点 $y(1)$ は、目標物 $t(2)$ 、もしくは対応する目標なし
- ・ 観測点 $y(2)$ は、目標物 $t(1)$ 、もしくは目標物 $t(2)$ 、もしくは対応する目標なし
- ・ 観測点 $y(3)$ は、目標物 $t(1)$ 、もしくは対応

する目標なし

- ・ 観測点 $y(4)$ は、目標物 $t(1)$ 、もしくは対応する目標なし

【0033】これを満たす目標物と観測点の組合せとしては、次のケースがある。

- ・ 目標物 $t(1)$ が観測点 $y(2)$ で、目標物 $t(2)$ が観測点 $y(1)$
- ・ 目標物 $t(1)$ が観測点 $y(2)$ で、目標物 $t(2)$ が対応なし
- ・ 目標物 $t(1)$ が観測点 $y(3)$ で、目標物 $t(2)$ が観測点 $y(1)$
- ・ 目標物 $t(1)$ が観測点 $y(3)$ で、目標物 $t(2)$ が観測点 $y(2)$
- ・ 目標物 $t(1)$ が観測点 $y(3)$ で、目標物 $t(2)$ が対応なし
- ・ 目標物 $t(1)$ が観測点 $y(4)$ で、目標物 $t(2)$ が観測点 $y(1)$
- ・ 目標物 $t(1)$ が観測点 $y(4)$ で、目標物 $t(2)$ が観測点 $y(2)$
- ・ 目標物 $t(1)$ が観測点 $y(4)$ で、目標物 $t(2)$ が対応なし
- ・ 目標物 $t(1)$ が対応なしで、目標物 $t(2)$ が観測点 $y(1)$
- ・ 目標物 $t(1)$ が対応なしで、目標物 $t(2)$ が観測点 $y(2)$
- ・ 目標物 $t(1)$ が対応なしで、目標物 $t(2)$ が対応なし

【0034】図4はこの組合せを探索木として表したものである。この探索木における各呼称は、図4では次の部分に対応しており、組合せは、根から葉までの各ノードの持つ状態に対応する。

- ・ 根 → Root
- ・ 節 → 木構造の分岐の中段に位置する、図4に楕円で示した部分
- ・ 葉 → 木構造の分岐の先端に位置する、図4に四角で示した部分
- ・ 枝 → 図4の根と節、節と節、節と葉を結ぶ線
- ・ ノード → 節と葉の両方を含む概念
- ・ 兄弟 → 根からノードまでに経由する枝数が等しいノード同士

【0035】以上を前提として、各部の動作を具体的に説明する。まず予測領域生成部2が、その内部に保持している目標物の情報(座標や速度など)から、観測点が観測された時間に対応する目標物 $t(1)$ および $t(2)$ の予測座標を算出する。また、同様に一定の存在確率以上となる、当該目標物 $t(1)$ 、 $t(2)$ の予測座標を中心とする予測領域(図2の楕円)を算出する。次にクラスタ生成部3が、その予測領域と観測点 $y(1) \sim y(4)$ との包含関係を調べる。複数の予測領域

域に含まれる観測点が存在する時、その観測点を包含する予測領域同士を結合してクラスタを形成する。図2に示す例では、観測点 $y(2)$ が2つの予測領域に含まれるため、この2つの予測領域を1つのクラスタとして結合する。

【0036】生成されたクラスタは節分割・統合部4に送られ、節分割・統合部4はそれより図4に示す探索木の節を生成する。負荷分散部9はこの節分割・統合部4が生成した節を取り込み、各タスク処理部5a~5dの負荷状況に応じて、その節の処理をタスク処理部5a~5dに分配する。ここで、タスク処理部5a~5dへ分配する節のレベルは、図4の観測点 $y(1)$ であっても、 $y(2)$ であっても、 $y(3)$ であっても構わない。なお、下位のレベルになるに従って、各節に対する処理量は通常小さくなる。各タスク処理部5a~5dは、この負荷分散部9によって分配された節から葉を算出し、各葉が示す組合せごとに信頼度を計算して、目標物と各観測点の対応関係についての重み付けを行う。

【0037】例えば、図4において、節10を受け取ったタスク処理部は、各組合せごとの信頼度として、以下の信頼度 $E(1) \sim E(3)$ を計算する。

$E(1) \rightarrow$ 観測点 $y(1)$ が対応なし、観測点 $y(2)$ が目標物 $t(2)$ 、観測点 $y(3)$ 、 $y(4)$ が対応なしのケースの信頼度

$E(2) \rightarrow$ 観測点 $y(1)$ が対応なし、観測点 $y(2)$ が目標物 $t(2)$ 、観測点 $y(3)$ が対応なし、観測点 $y(4)$ が目標物 $t(1)$ のケースの信頼度

$E(3) \rightarrow$ 観測点 $y(1)$ が対応なし、観測点 $y(2)$ が目標物 $t(2)$ 、観測点 $y(3)$ が目標物 $t(1)$ 、観測点 $y(4)$ が対応なしのケースの信頼度

【0038】次に当該タスク処理部5a~5dは、計算した信頼度の値をもとに、図4に示した節10での目標物と各観測点の対応関係について、以下の重み付けを行う。

・ 目標物 $t(1)$ が観測点 $y(3)$ に対応する重みを、 $E(3)$ ・ 目標物 $t(1)$ が観測点 $y(4)$ に対応する重みを、 $E(2)$ ・ 目標物 $t(1)$ が対応なしとなる重みを、 $E(1)$

・ 目標物 $t(2)$ が観測点 $y(2)$ に対応する重みを、 $E(1) + E(2) + E(3)$

【0039】次に節分割・統合部4は、各タスク処理部5a~5dが算出した「目標物と各観測点の対応関係についての重み」を集計する。例えば、目標物 $t(2)$ であれば、観測点 $y(1)$ に対応する重み、観測点 $y(2)$ に対応する重み、および対応なしとなる重みが得られる。

【0040】目標物航跡更新部6はこの節分割・統合部4が統合した後の情報を受け取り、目標物の位置を算出して、目標物の座標や速度等の内部情報の更新を行う。例えば、目標物 $t(2)$ であれば、上記3つの重み付け

に基づいて目標物 $t(2)$ の座標を求める。なお、この重み付けに基づいた計算結果を、当該目標物の座標と見做す。以上で各部の動作が終了する。

【0041】次に、タスク処理部5a~5dにおける、それぞれの負荷状況に応じた節の分配について説明する。まず、負荷分散部9は、節分割・統合部4が生成した探索木のあるレベルの全ての節の処理を均等に各タスク処理部5a~5dに分配する。均等に節の処理を分配する方法には、節の数を均等にする、あるいは、節の下位の葉の数を均等にするなど考えられる。

【0042】次に、負荷分散部9が転送線8gを介して各タスク処理部5a~5dに負荷の報告要求を出し、各タスク処理部5a~5dはその要求に応じて負荷状況を負荷分散部9に報告する。ここで、負荷状況には、オペレーティングシステムで管理しているジョブ・キューの単位時間当たりの長さ、すなわち、ジョブ・キューに入っているジョブの数、あるいは、各タスク処理部5a~5dがまだ処理していない節や葉の数などが考えられる。

【0043】次に、負荷分散部9は各タスク処理部5a~5dからの負荷状況の報告を受け取ると、その情報をもとに、各タスク処理部5a~5dの負荷が均等となるように、各タスク処理部5a~5dに節の移動を要求する。ここで、例えば負荷状況を各タスク処理部5a~5dの未処理の節の数とした場合、各タスク処理部5a~5dからの負荷状況の報告が、それぞれ3、5、4、4であったものとする。これより、全てのタスク処理部5a~5dにおける未処理の節の数は $3+5+4+4=16$ なので、4つのタスク処理部5a~5dの負荷状態を均等化した場合、各タスク処理部5a~5dの担当すべき節の数は4つずつとなる。従って、負荷分散部9は転送線8gを介して、タスク処理部5bに1つの節の処理をタスク処理部5aに移動するように指示する。これにより、各タスク処理部5a~5dの未処理の節の数が4つずつと等しくなる。この負荷状況の収集とそれによる各タスク処理部5a~5d間の節の移動を定期的に行う。

【0044】以上のように、この実施の形態1によれば、負荷分散部9が、各タスク処理部5a~5dの負荷状況に応じて、節の処理をタスク処理部5a~5dに分配しているので、各タスク処理部5a~5dが汎用のワークステーションのように他の処理が行われているマシン上で動作する場合でも、各タスク処理部5a~5dの処理時間が均等になり、最適な負荷分散を行うことが可能になるという効果が得られる。

【0045】また、各タスク処理部5a~5dの未処理の節の数が定期的に均等化されるので、節の処理以外の処理がタスク処理部5a~5dと同一のプロセッサ上で行われていても、各タスク処理部5a~5dの全ての節の処理がほぼ同じ時間で終了し、最適な負荷分散が行え

るという効果も得られる。

【0046】実施の形態2。以上の実施の形態1では、負荷分散部9が負荷分散のタイミングの主導権を取って、定期的に各タスク処理部5a～5dの負荷を均等化することにより、各タスク処理部5a～5dにおける節の処理をほぼ同じ時間で終了させるものについて説明したが、タスク処理部5a～5dが負荷分散のタイミングの主導権を取り、負荷分散部9に対して仕事の転送要求を行うようにしてもよい。この実施の形態2はそのような並列多目標追尾装置について述べたもので、その構成は図1に示した実施の形態1の場合と同様であるため、ここではその説明は省略する。

【0047】次に動作について説明する。なお、ここでは実施の形態1とは異なる部分についての説明を行う。まず、負荷分散部9は、節分割・統合部4が生成した探索木の、あるレベルの節の処理を1単位ずつ各タスク処理部5a～5dに分配する。各タスク処理部5a～5dではその探索木の節を受け取るとその処理を開始する。各タスク処理部5a～5dは受け取った節の処理が終了して負荷がなくなると、負荷分散部9に対して次の仕事、すなわち処理する節の転送を要求する。負荷分散部9はその転送要求を受け取ると、当該要求を出したタスク処理部5a(～5d)に対して1単位の節の処理(負荷)を与える。以上の処理を全ての節がなくなるまで繰り返す。

【0048】以上のように、この実施の形態2によれば、各タスク処理部5a～5dは仕事なくなると次の仕事の転送要求を負荷分散部9へ出すので、節の処理以外の処理が同一マシン上で行われていても、各タスク処理部5a～5dの全ての節の処理がほぼ同じ時間で終了し、最適な負荷分散が行えるという効果が得られ、さらに、1つの節の処理に対応する処理量が小さい場合でも、定期的な負荷状況の収集処理が必要ないので、負荷分散のオーバーヘッドが少なくなるという効果も得られる。

【0049】実施の形態3。上記各実施の形態では、各タスク処理部5a～5dは1つの節の処理が完了してから次の節の処理の転送要求を負荷分散部9に出す場合について説明したが、タスク処理部5a～5dに仕事をある程度確保しておき、それを順次処理してゆくようにしてもよい。この実施の形態3はそのような並列多目標追尾装置について述べたもので、その構成は図1に示した実施の形態1の場合と同様であるため、ここではその説明は省略する。

【0050】次に動作について説明する。なお、この場合にも実施の形態1とは異なる部分についての説明を行う。まず、負荷分散部9は最初に、節分割・統合部4が生成した探索木の、あるレベルの節の処理を複数単位ずつ各タスク処理部5a～5dに分配する。各タスク処理部5a～5dは、この複数単位の節を受け取ると、その内

の1単位を除く残りの節を保持し、保持していない残りの1単位の節の処理を実行する。各タスク処理部5a～5dは当該節の処理が終了すると、負荷分散部9に対して次の仕事、すなわち次に処理すべき節の転送を要求する。そしてこの転送要求に対する応答が負荷分散部9より返される前に、保持しておいた節の内の1単位の節を取り出してその処理を開始する。負荷分散部9はタスク処理部5a(～5d)から次に処理する節の転送要求を受け取ると、当該要求を送出したタスク処理部5a(～5d)に対して、次の節の処理の分配を行う。また、要求を出したタスク処理部5a(～5d)は、負荷分散部9から次に処理する節の分配を受けると、それを保持する。以上の処理を全ての節がなくなるまで繰り返す。

【0051】以上のように、この実施の形態3によれば、各タスク処理部5a～5dは仕事なくなると、負荷分散部9に対して次の仕事の転送要求を出すとともに、保持しておいた節の処理をすぐに開始しているので、負荷分散部9へ次の仕事の転送要求を出してからその応答を受け取るまでの間も、他の節の処理が実行されていて、タスク処理部5a～5dが節の処理を行っていない時間がほとんどなくなるため、最適な負荷分散が行えるという効果が得られる。

【0052】実施の形態4。上記各実施の形態では、負荷分散の単位を一定としているものについて説明したが、負荷分散の単位を時間とともに小さくするようにしてもよい。この実施の形態4はそのような並列多目標追尾装置について述べたもので、その構成は図1に示した実施の形態1の場合と同様であるため、ここではその説明は省略する。

【0053】ここで、並列多目標追尾装置における負荷分散の単位は節であるが、その処理量に相当するサイズは、例えば節の下位の葉の数で測ることができる。図4に示した探索木からも分かるように、葉の数は節のレベルが上がるほど増加し、レベルが下がるほど減少する。従って、負荷分散に際して、節を分解してよりレベルの低い節を負荷分散の単位とすることにより、より小さなサイズの仕事を生成することは容易である。

【0054】ここで、負荷分散の単位のサイズは、小さければ小さいほど負荷を分散する回数が増えるため負荷分散のためのオーバーヘッドが大きくなるが、処理の終了を各タスク処理部5a～5dで均一化するという面では小さいほどよい。従って、最初は高いレベルの節を負荷分散の単位とし、処理が進むにつれて節を分解し、より低いレベルの節を負荷分散の単位とすることで、負荷分散オーバーヘッドを抑えた上で、各タスク処理部5a～5dの節の処理の終了を均一化し、最適な負荷分散を行うことができる。

【0055】なお、時間とともに負荷分散の単位を小さくする場合、実施の形態1のように各タスク処理部5a～5dが仕事(すなわち節の処理)を保持しているとき

には、各タスク処理部5a～5dで節を分解する。また実施の形態2のように負荷分散部9が節の処理を保持しているときには、負荷分散部9で節を分解し、実施の形態3のように負荷分散部9および各タスク処理部5a～5dで節の処理を保持しているときには、負荷分散部9および各タスク処理部5a～5dの両方で節を分解する。

【0056】以上のように、この実施の形態4によれば、時間とともに負荷分散の単位を小さくしているのので、負荷分散のオーバーヘッドを抑えた上で、各タスク処理部5a～5dの節の処理の終了を均一化し、最適な負荷分散を行うことができるという効果が得られる。

【0057】実施の形態5。上記実施の形態4では、時間とともに負荷分散の単位を小さくする場合について説明したが、時間によらずに、1つの負荷分散単位の処理時間が一定以上長くなった場合に負荷分散単位を小さくするようにしてもよい。この実施の形態5はそのような並列多目標追尾装置について述べたもので、その構成は図1に示した実施の形態1の場合と同様であるため、ここではその説明は省略する。

【0058】探索木の同じレベルの節の処理を負荷分散の単位としている場合、節によってはその下位の葉の数は均等ではない。従って、ある節の下位の葉の数が非常に多い場合は、その節の処理に非常に長い時間がかかり、最悪の場合、他の節の処理は全て完了しているのに、その節の処理だけが残ってしまうということがあり得る。

【0059】そこで、各タスク処理部5a～5dでは節の処理時間を監視しており、ある節の処理を行っている最中に、その節の処理を開始してからの実行時間が一定時間以上経過した場合、その節の下位のまだ処理していない節群の一部を負荷分散部9に返す。そして実施の形態1の場合のように、各タスク処理部5a～5dが仕事（節の処理）を保持している場合には、次の負荷分散のタイミングまで返された節群を保持しておき、次の負荷分散の際に負荷の低いタスク処理部5a～5dに対してその節群の処理を分配する。また、実施の形態2の場合のように負荷分散部9が節の処理を保持している場合には、返された節群をそのまま保持して既に保持している節と同様に扱い、実施の形態3の場合のように負荷分散部9および各タスク処理部5a～5dで節の処理を保持している場合も同様に、返された節群をそのまま保持して既に保持している節と同様に扱う。

【0060】以上のように、この実施の形態5によれば、処理時間の非常にかかる節の処理の一部を負荷分散部9へ返すことにより、処理能力に余裕のある他のタスク処理部5a～5dへこれらの節群の処理を分散しているので、負荷が分散され、全体として処理が早く終了するようになり、最適な負荷分散が行えるという効果が得られる。

【0061】実施の形態6。上記実施の形態5では、タスク処理部5a～5dが未処理の節群の処理を負荷分散部9に返すと判断する1つの負荷分散単位の処理時間の上限値を、一定にしたものについて説明したが、当該上限値を時間とともに小さくするようにしてもよい。この実施の形態6はそのような並列多目標追尾装置について述べたもので、その構成は図1に示した実施の形態1の場合と同様であるため、ここではその説明は省略する。

【0062】タスク処理部5a～5dが未処理の節群の処理を負荷分散部9へ返す場合、返すと判断する1つの負荷分散単位の処理時間の上限値は、それが大きいと、負荷分散部9へ返すと判断した時には既に他のタスク処理部5a～5dの処理が完了している危険性があり、小さいと他のタスク処理部5a～5dの処理が当分完了しないのにも係わらず未処理の節群の処理が負荷分散部9へ返されることになるため、無駄な仕事の移動になるという危険性がある。

【0063】そこで、最初のうちは、この未処理の節群の処理を負荷分散部9へ返すと判断する上限値を大きくしておき、時間とともにそれを小さくしていく。それにより、最初のうちは大きな上限値により、無駄となる負荷分散部9への節群の返却を抑えることができる。また、全体の処理の最後の方では、小さな上限値により他のタスク処理部5a～5dが処理を完了する前に、処理に時間がかかっているタスク処理部5a～5dの仕事の一部を他のタスク処理部5a～5dへ分散させることができ、最適な負荷分散とすることができる。

【0064】以上のように、この実施の形態6によれば、負荷分散部9に節群を返却する際の負荷量の上限値を時間とともに小さくしているので、最初のうちは無駄となる節群の返却を抑えることができ、処理の最後の方では、処理に時間がかかっているタスク処理部5a～5dの仕事の一部を他のタスク処理部5a～5dへ確実に分散させることができるようになり、最適な負荷分散を行うことが可能になるという効果が得られる。

【0065】実施の形態7。上記各実施の形態では、各タスク処理部5a～5dと負荷分散部9とを異なるプロセッサ上で動作させたものについて説明したが、複数のタスク処理部5a～5dの内の1つを負荷分散部9と同一のプロセッサ上で動作させるようにしてもよい。節の処理を行っている際の負荷は、タスク処理部5a～5dで最も高く、負荷分散部9にはほとんど負荷がない。従って、負荷分散部9をタスク処理部5a～5dの1つと同一プロセッサ上で動作させることにより、全てのプロセッサの負荷を高い状態とすることができる。

【0066】以上のように、この実施の形態7によれば、タスク処理部5a～5dの1つと同一プロセッサ上で負荷分散部9を動作させているので、全てのプロセッサでその負荷を高い状態とすることができ、無駄のない並列処理が可能となるという効果が得られる。

【0067】

【発明の効果】以上のように、この発明によれば、負荷分散部を設け、クラスタについて生成された探索木の節を複数のタスク処理部に分配して処理する際に、各タスク処理部の負荷状況に応じて節の処理を分配するように構成したので、各タスク処理部が、他の処理を実行しているプロセッサ上で動作するような場合でも、各タスク処理部の処理時間を均等にすることができ、最適な負荷分散を行うことが可能な並列多目標追尾装置が得られるという効果がある。

【0068】この発明によれば、各タスク処理部の負荷状況を定期的に監視して、負荷が少なくなると、負荷の多いタスク処理部から節の処理を移動させるように構成したので、各タスク処理部の未処理の節の数が定期的に均等化され、節の処理以外の処理がタスク処理部と同一のプロセッサ上で行われていても、各タスク処理部の全ての節の処理がほぼ同じ時間で終了し、最適な負荷分散を行うことができるという効果がある。

【0069】この発明によれば、各タスク処理部に対して節の処理を1単位ずつ分配し、タスク処理部は分配された節の処理が終了すると、次の節の分配を要求して新たな1単位の節の処理の分配を受けるように構成したので、同一のプロセッサ上で節の処理以外の処理が行われていても、各タスク処理部の全ての節の処理がほぼ同じ時間で終了し、最適な負荷分散が可能になるとともに、1つの節の処理に対応する処理量が小さい場合でも、定期的な負荷状況の収集処理が必要ないので、負荷分散のオーバーヘッドが少なくなるなどの効果がある。

【0070】この発明によれば、各タスク処理部に対して負荷分散部より、最初に複数単位ずつの節の処理を分配し、各タスク処理部ではその内の1単位の節の処理を開始して、残りを保持しておき、当該節の処理が終了すると、次の節の分配を負荷分散部に要求するとともに、その要求に対する応答が返される前に保持しておいた次の1単位の節の処理を開始するように構成したので、各タスク処理部は仕事が終わると次の仕事の転送要求を負荷分散部に対して出すとともに、保持しておいた節の処理がすぐに開始されるため、負荷分散部への節の転送要求を出してから対応する節の転送を受けるまでの間も、保持されている他の節の処理が実行されて、タスク処理部が節の処理を行っていない時間がほとんどなくなり、最適な負荷分散が可能になるという効果がある。

【0071】この発明によれば、負荷分散の単位を時間とともに小さくしていくように構成したので、負荷分散のオーバーヘッドを抑えた上で、各タスク処理部における

節の処理の終了を均一化することができ、最適な負荷分散が可能になるという効果がある。

【0072】この発明によれば、1つの負荷分散単位の処理時間が一定以上になると、処理している節の下位の一部の節群の処理を負荷分散部へ返すように構成したので、負荷分散により、処理能力に余裕のある他のタスク処理部にそれらの節群の処理を分配することができ、全体として処理が早く終了するようになるため、最適な負荷分散が可能になるという効果がある。

【0073】この発明によれば、一部の節群の処理を負荷分散部へ返すと判断する際の処理時間の上限値を、時間とともに小さくしていくように構成したので、最初のうちは大きな上限値により、無駄となる負荷分散部への節群の返却を抑えることができ、全体の処理の最後の方では小さな上限値により、他のタスク処理部の仕事が完了する前に処理に時間のかかっているタスク処理部の仕事の一部を他のタスク処理部へ分散させることができるため、最適な負荷分散が可能になるという効果がある。

【0074】この発明によれば、同一のプロセッサ上でタスク処理部の1つと負荷分散部とを動作させるように構成したので、全てのプロセッサでその負荷を高い状態とすることができ、無駄のない並列処理が可能となるという効果がある。

【図面の簡単な説明】

【図1】 この発明による並列多目標追尾装置の構成を示すブロック図である。

【図2】 この発明の並列多目標追尾装置におけるJPD Aによる目標物と観測点の関係を示す説明図である。

【図3】 この発明の並列多目標追尾装置における目標物と観測点の関係を表した行列 Ω を示す説明図である。

【図4】 この発明の並列多目標追尾装置における目標物と観測点の組合せを表した探索木を示す説明図である。

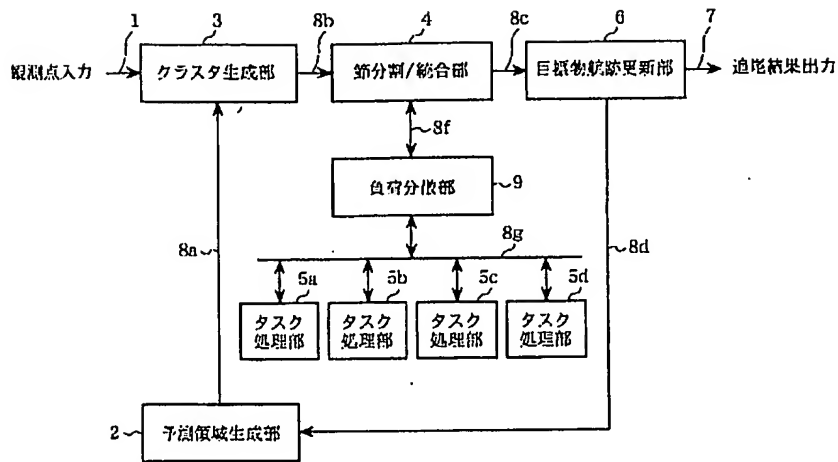
【図5】 従来の並列多目標追尾装置における追尾処理のタスクグラフを示す説明図である。

【図6】 従来の並列多目標追尾装置における追尾処理のプロセッサグラフを示す説明図である。

【符号の説明】

1 入力線、2 予測領域生成部、3 クラスタ生成部、4 節分割・統合部、5 a～5 d タスク処理部、6 目標物航跡更新部、7 出力線、8 a～8 d、8 f、8 g 転送路、9 負荷分散部、10 節、t (1)、t (2) 目標物、y (1)～y (4) 観測点。

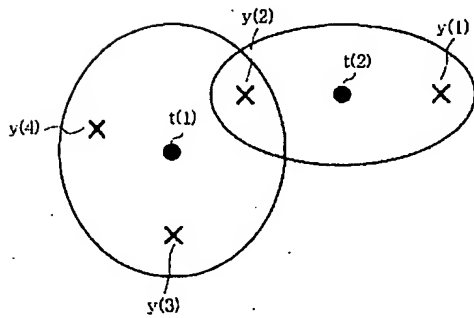
【図1】



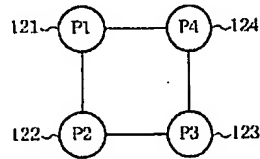
【図3】

$$\begin{array}{c}
 t \\
 \begin{array}{ccc}
 0 & 1 & 2
 \end{array} \\
 \left. \begin{array}{ccc}
 1 & 0 & 1 \\
 1 & 1 & 1 \\
 1 & 1 & 0 \\
 1 & 1 & 0
 \end{array} \right\} y
 \end{array}$$

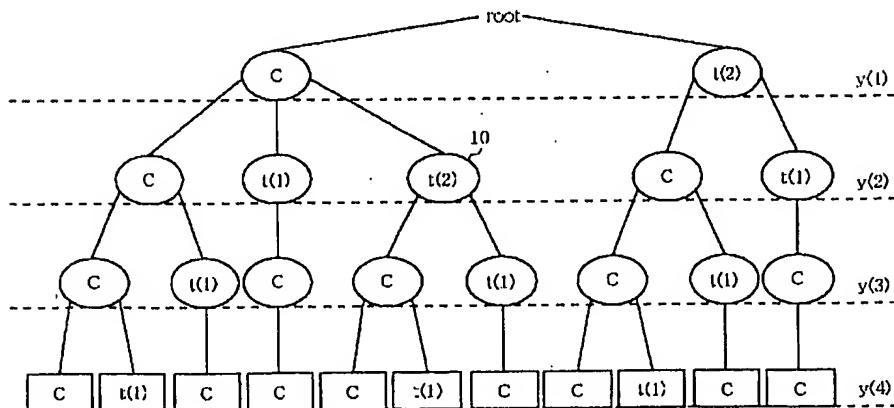
【図2】



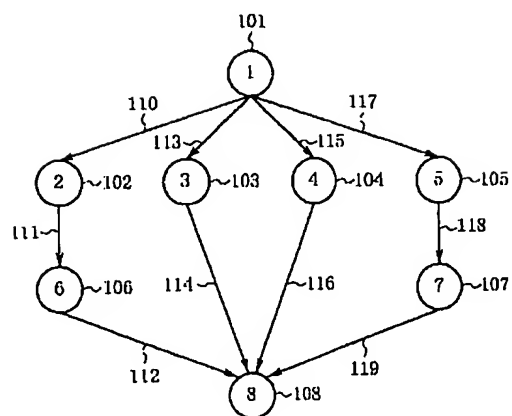
【図6】



【図4】



【図5】



**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☒ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.